CLAIMS

What is claimed is:

1. A computer-implemented method for dynamic instrumentation of an executable application program using an instrumentation program, the application program including a plurality of original functions, each original function having an entry point and an endpoint, comprising:

creating a shared memory segment for the instrumentation program and the application program;

upon initial invocation of the original functions in the application program, creating in the shared memory segment corresponding substitute functions including instrumentation code; and

executing the substitute functions in lieu of the original functions in the application program.

2. The method of claim 1, further comprising:

patching the function entry points with breakpoint instructions; and

creating the substitute functions upon encountering the breakpoint instructions.

3. The method of claim 2, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions.

4. The method of claim 3, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising:

after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function.

5. The method of claim 1, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising:

after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function.

6. The method of claim 1, further comprising:

copying a segment of the executable application program to selected area of memory by the instrumentation program;

replacing the segment of the application program with code that allocates the shared memory by the instrumentation program;

executing the code in the application program that allocates the shared memory segment; and

restoring the segment of the executable application from the selected area of memory to the application program by the instrumentation program after the shared memory is allocated.

7. The method of claim 6, further comprising:

patching the function entry points with breakpoint instructions; and

creating the substitute functions upon encountering the breakpoint instructions.

8. The method of claim 7, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions.

9. The method of claim 8, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising:

after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function.

10. The method of claim 6, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising:

after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function.

11. The method of claim 6, wherein the executable application program includes a plurality of threads and further comprising:

before the step of copying the segment of the executable application program suspending all threads of the executable application program, and selecting one of the suspended threads; and

after replacing the segment of the executable application program with the code that allocates the shared memory, resuming execution of the one of the suspended threads at the code that allocates the shared memory.

12. The method of claim 11, further comprising:

patching the function entry points with breakpoint instructions; and

creating the substitute functions upon encountering the breakpoint instructions.

13. The method of claim 12, further comprising replacing the break instruction at the entry points of the functions in the application program with branch instructions that target the substitute functions.

14. The method of claim 13, wherein the executable application program includes one or more branch instructions having target addresses that reference entry points of one or more of the original functions, further comprising:

after creating a substitute function corresponding to an original function, for a branch instruction that references the original function replacing the target addresses to reference the substitute function.

15. An apparatus for dynamic instrumentation of an executable application program by an instrumentation program, the application program including a plurality of original functions, each original function having an entry point and an endpoint, comprising:

means for creating a shared memory segment for the instrumentation program and the application program;

means for creating in the shared memory segment corresponding substitute functions including instrumentation code upon initial invocation of the original functions in the application program; and

means for executing the substitute functions in lieu of the original functions in the application program.